Title of the Prior Art

Japanese Published Patent Application No. Hei.11-282756, Date of Publication: October 15, 1999

Concise Statement of Relevancy

A microprocessor which stores a specific secret key in a built-in storage register executes an application program which is encrypted using a working key corresponding to a secret key of a symmetric cryptography system by an unencrypted startup program, and checks as to whether a permit code which is obtained by encrypting the working key for only the application program by a public key is present on a hard disk or not. When the check is completed, the work key which is obtained by decoding the permit code with the secret key stored in the storage register is stored in the storage register, and the application program is decoded by the work key to be executed. Therefore, the application program can be executed by only the specific microprocessor. Further, since the secret key does not appear outside, the encrypted application program can be distributed on the market through duplicable media.

THIS PAGE IS BLANK

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号

特開平11－282756

(43)公開日　平成11年(1999)10月15日

(51)Int.Cl.[6]		識別記号	FI		
G06F	12/14	320	G06F	12/14	320A
		310			310Z
	9/06	550		9/06	550Y
					550Z

審査請求　未請求　請求項の数5　FD　（全 18 頁）

(21)出願番号　特願平10－103957

(22)出願日　平成10年(1998)３月31日

(71)出願人　000110468
ナカミチ株式会社
東京都小平市鈴木町１丁目153番地

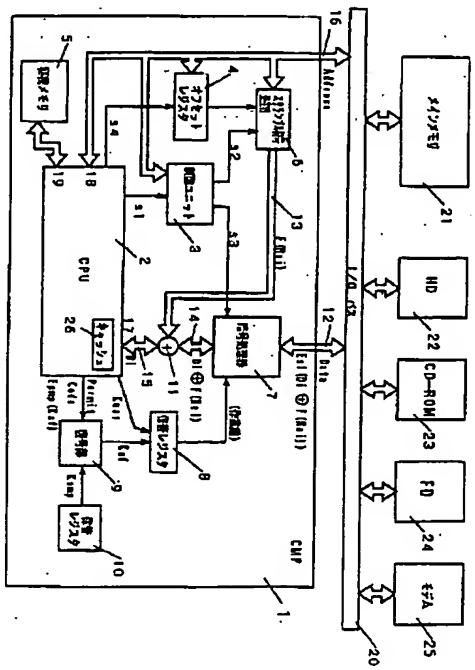
(72)発明者　末松　俊成
東京都小平市鈴木町１丁目153番地　ナカミチ株式会社内

(54)【発明の名称】　秘密データ管理方法

(57)【要約】

【目的】コンピュータの秘密データを管理することによりソフトウエアに対する不正行為を防止可能とし、またその保管を円滑に処理する管理方法を提供する。

【構成】秘密データを保持するための秘密メモリ５を有するマイクロプロセッサだどの信号処理手段において、この秘密メモリに書き込まれた秘密データの外への読み出しを一切禁止し、この秘密データと比較する参照データとが一致するかを判定する。更に、この秘密データを更新又は消去する際には、対象の秘密データと同一のデータを用意して初めて可能とする。また、パスワードを秘密データに対応して記憶することにより、このパスワードが秘密データのアドレスの代用となる。



1

【特許請求の範囲】

【請求項１】秘密データを保存するための秘密メモリを有する信号処理手段において、該秘密メモリに書き込まれた前記秘密データの前記信号処理手段外への読み出しを禁止し、参照データを得て該参照データと前記秘密データとの一致を判定することを特徴とする秘密データ管理方法。

【請求項２】秘密データを保存するための秘密メモリを有する信号処理手段において、該秘密メモリに書き込まれた前記秘密データの前記信号処理手段外への読み出しを禁止し、記録済み領域の所望のデータを書換える際には、別途該所望のデータに対応する参照データと更新データとを得て、前記所望のデータと前記参照データとが一致したとき、前記所望のデータと前記更新データとの書換えを実行するようにしたことを特徴とする秘密データ管理方法。

【請求項３】秘密データを保存するための秘密メモリを有する信号処理手段において、該秘密メモリに書き込まれた前記秘密データの前記信号処理手段外への読み出しを禁止し、記録済み領域の所望のデータを消去する際には、別途該所望のデータに対応する参照データを得て、前記所望のデータと前記参照データとが一致したとき、前記所望のデータの消去を実行するようにしたことを特徴とする秘密データ管理方法。

【請求項４】マイクロプロセッサ内に秘密メモリを有し、該秘密メモリに実行するアプリケーションソフトに対応して設定された識別コードを保存し、前記アプリケーションソフトの実行が終了する毎に生成される秘密値を該アプリケーションソフトの関連ファイルに書き込んで外部メモリに保存すると共に、前記識別コードによって識別可能に前記秘密メモリに保存し、再度前記アプリケーションソフトを実行する際にはそれぞれ保存された前記両秘密値を照合するようにしたことを特徴とする秘密データ管理方法。

【請求項５】秘密データを保存するための秘密メモリにおいて、該秘密メモリは前記秘密データに識別コードを対応して保存し、前記秘密データを記録または再生する場合には、前記メモリ内に保存された前記識別コードに基づいて前記秘密データを選択することを特徴とする秘密データ管理方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、記録するデータの機密性を高めるためのメモリのデータ管理方法に関し、特にコンピュータの秘密データを管理することにより、ソフトウエアに対する不正行為を防止可能とし、且つこれ等秘密データの保管を円滑に処理する管理方法に関する。

【０００２】

【従来の技術】従来、この種の秘密データの管理方法としては、ハードディスクの不良セクタの管理システムを利用し、仮想の不良セクタにこの秘密データを保管する方法などがあった。

2

【０００３】

【発明が解決しようとする課題】しかしながら、このようなデータ管理方法においては、専門的知識を有する人であれば、これ等の秘密データを改ざんすることが可能であるため、もっと安全性の高いスシテムを構築する必要があった。

【０００４】

【課題を解決するための手段】秘密データを保持するための秘密メモリを有するマイクロプロセッサなどの信号処理手段において、この秘密メモリに書き込まれた秘密データの外への読み出しを一切禁止し、この秘密データと比較する参照データとが一致するかを判定する。

【０００５】更に、この秘密データを更新又は消去する際には、対象の秘密データと同一のデータを用意して初めて可能とする。また、パスワード等の識別コードを秘密データに対応して保存することにより、この識別コードが秘密データのアドレスの代用となる。

【０００６】更に、実行するアプリケーションソフトに対応して設定されたパスワード等の識別コードを保持し、前記アプリケーションソフトの実行が終了する毎に生成される秘密値を該アプリケーションソフトの関連ファイルに書き込んで外部メモリに保存すると同時に、前記識別コードによって識別可能に前記秘密メモリに保存し、再度前記アプリケーションソフトを実行する際には前記両秘密値を照合する。

【０００７】

【発明の実施の形態】図１は、本発明の一実施例を示す構成図で、図中１は、暗号化機能を有するマイクロプロセッサ（以下ＣＭＰと称す）であり、１チップで構成されている。この中の２は中央演算処理装置（以下ＣＰＵと称す）を示し、３は制御ユニットを示す。この制御ユニット３は、ＣＰＵ２からの指令信号ｓ１を入力して、スクランブル符号発生器６の動作のオン、オフを指令する動作指令信号ｓ２と暗号処理器７の暗号化／復号化機能の実行、停止の各動作を指令する動作指令信号ｓ３とをそれぞれ出力する。

【０００８】このＣＭＰ１は、固有の秘密鍵Ｋsmpを所有する。これは内蔵の保管レジスタ１０に保管され、後述するごとくこの秘密鍵に対応する公開鍵Ｋpmpで作業鍵Ksfを暗号化したパーミットコードを復号部９で復号化するための鍵となる。この秘密鍵Ｋsmpは、ＣＭＰの製造元のみが知るもので、ＣＭＰの外に現れないようにされてユーザー等の第３者が知ることは出来ず、逆に公開鍵Ｋpmpは、ＣＭＰの入手時にユーザーに知らされて全ての人が知り得るものである。ここで復号化された作業鍵Ｋsfは、保管レジスタ８に保管される。

【０００９】ＣＭＰ１につながるアドレスバス１６とデ

3

ータバス１２は、Ｉ／Ｏバス２０を介してメインメモリ２１、ハードディスク２２、ＣＤ－ＲＯＭ２３、フロッピディスク２４の各ドライブ、及びモデム２５の各入出力ポートにつながっている。

【００１０】一方ＣＭＰ１内部において、アドレスバス１６は、ＣＰＵ２のアドレスポート１８、制御ユニット３、オフセットレジスタ４、及びスクランブル符号発生器６につながり、データバス１２は、暗号処理器７、加算回路１１を介してＣＰＵ２のデータポート１７につながっている。

【００１１】オフセットレジスタ４は、ＣＰＵ２から出力される取り込み信号ｓ４を受信し、後述するようにＣＰＵ２がメインメモリ２１との間で読み書きする際のデータの先頭位置を示す先頭アドレスデータを取り込んでメモリし、この先頭アドレスデータをスクランブル符号発生器６に出力する。スクランブル符号発生器６は、後述するようにこの先頭アドレスデータとアドレスバス１６から入力するアドレスデータをもとに生成したスクランブル符号Ｆ(Re)を信号経路１３を介して加算器１１に出力する。

【００１２】加算器１１は、このスクランブル符号Ｆ(Re)とデータ経路１４，１５の一方のデータ経路から入力するデータの排他論理和演算を行い、その結果を他方のデータ経路に出力することによりこれ等のデータのスクランブルとデスクランブルを実行する。

【００１３】ここで、図１の各部で、データに対して行う暗号化、復号化、及びデータスクランブルの各内容について更に説明する。ここで使用される共通鍵方式と公開鍵方式の２つの暗号化方式は一般的なものであり、その詳細な説明は省略するが、本文で用いるそれらの記述方法について説明する。

【００１４】共通鍵方式では、データＤを暗号化するときに秘密鍵Ｋabを用い、これによって暗号化されたデータをＥab（Ｄ）と標記する。この場合、秘密鍵Ｋabがあれば、Ｅab（Ｄ）と記述された暗号化データを復号化して再びデータＤを取り出すことができる。

【００１５】一方、公開鍵方式では、データＤを暗号化するときに、公にされた公開鍵Ｋpabを用いて行ない、これによって暗号化された暗号化データをＥpab（Ｄ）と標記する。この場合、この暗号化データの解読はこの公開鍵Ｋpabに対応する秘密鍵Ｋsabによってのみ行なうことが出来る。

【００１６】即ち、共通鍵方式では、一つの秘密鍵Ｋabによってデータの暗号化と復号化を行なうが、公開鍵方式では、公にする公開鍵Ｋpabではデータの暗号化のみが可能であって、その復号化には秘密鍵Ｋsabを用いて行なう。

【００１７】次に、アドレスによって行なわれるデータスクランブルについて、その一例を示す本実施例の原理を説明する。簡単のため、データＤが記憶手段に書き込まれる際には１バイト単位で行なわれ、各単位毎にアドレスＡdを対応させて処理するものとする。また暗号化スクランブルされるデータは、メインメモリ２１内の連続した領域におかれ、この一連の連続するデータの集合をデータ群と呼ぶことにする。１つのプログラムやデータは、通常１つ以上のデータ群で構成されるが、スクランブルは、データ群毎に処理される。

4

【００１８】以降では説明を簡単にするため、プログラムやデータは、１データ群で構成されるものと仮定する。そこで、データ群ＤＸを１バイトデータのデータ列、Ｄ1，Ｄ2……ＤＮの集合として標記し、アドレス群ＡdＸをこれ等の各１バイトデータに対応して付されるアドレスＡd1，Ａd2……ＡdＮの集合として標記する。

【００１９】従って、この連続データの各アドレスＡdi（i=1，2，……Ｎ）は、その先頭アドレスをＴoとし（通常はＴo＝Ａd1）、この先頭アドレスからの相対位置を示す相対アドレスをＲeiとしたとき、Ａdi＝Ｔo＋Ｒeiとして表せる。図１のオフセットレジスタ４は、この先頭アドレスＴoをメモリするものである。

【００２０】一方、スクランブル符号発生器６は、逐次入力するアドレスＡdiと先頭アドレスＴoとからこの相対アドレスＲeiを算出し、この値に１対１で対応するも、規則性を示さない１バイトのスクランブル符号Ｆ(Ｒei)を発生する。このようにして生成されるスクランブル符号の集合をＦ(ＲeＸ)と標記する。加算器１１は、データ経路１４，１５の内、一方から逐次入力する１バイトのデータと、このデータのアドレスに対応して入力するスクランブル符号Ｆ(Rei)との排他論理和データを他方のデータ経路に出力する。

【００２１】従って、例えば、ＣＰＵ２から出力されるデータＤ1がメインメモリ２１にメモリされる時の一過程を説明すると、この時にデータＤ1、例えば（10010001）が出力されると共に、このデータの番地を指定するアドレスＡd1が出力される。加算器１１は、このデータを入力すると共に、この時の指定アドレスから算出された相対アドレスＲe1に対応して発生したスクランブル符号Ｆ(Re1)、例えば(11001100)を入力し、これ等の排他論理和(01011101)を暗号処理器７に出力する。

【００２２】このように、データＤiとスクランブル符号Ｆ(Rei)との排他論理和によってスクランブルされたデータを（Ｄi(+)Ｆ(Rei)）と記述して図１中に示す。従って、この図からもわかるように、データ経路１４はスクランブルされたデータの経路となる。

【００２３】次に、スクランブルされたデータを解除する例として、ＣＰＵ２が上記したデータＤ1を読み込む時の一過程を考えてみる。この時、このデータＤ1を読み出すためのデータとして上記アドレスＡd1が出力され、これに伴ってスクランブル符号Ｆ(Re1)がスクランブル符号発生器６から出力されことが理解される。

【００２４】従って加算器１１は、この時データ経路１

5

４に表れるスクランブルされた（D1(+)F(Re1)）のデータ列(01011101)とスクランブル符号F(Re1)のデータ列(11001100)との排他論理和(10010001)、即ちデスクランブルされる前の元のデータD1をＣＰＵ２に出力する。

【００２５】このように、加算器１１は、データD1とこれに対応するスクランブル符号F(Re1)との排他論理和をとってスクランブルされたデータ（D1(+)F(Re1)）を形成し、逆にこのデータ（D1(+)F(Re1)）とスクランブル符号F(Re1)との排他論理和をとってデスクランブルされる前の元のデータD1を生じるべく動作することが理解される。

【００２６】次に、図２の実行準備フローに従って、暗号化され更にスクランブルされたアプリケーションソフトを入手し、その実行に際してＣＰＵ2内への読み込みを可能にするまでの準備過程について主に説明する。この時のアプリケーションソフトをデータ群DapXで示すと、上記した如くこのデータ群DapXに対応して生成されるスクランブル符号F(ReX)でスクランブルされ、共通鍵方式の作業鍵Ksf２で暗号化されたアプリケーションソフトは、Esf２（DapX(+)F(ReX)）で示される。

【００２７】尚、この作業鍵Ksf２は、前記した共通鍵方式の秘密鍵に相当するものであるが、公開鍵方式の秘密鍵と区別するために以後作業鍵と称すことにする。また、ここで設定される作業鍵Ksf２は、このアプリケーションソフト供給元がこのソフト専用に用意するもので、この作業鍵Ksf２で他のアプリケーションソフトを解読することは出来ない。

【００２８】この暗号化されたアプリケーションソフトEsf２（DapX(+)F(ReX)）は、図１０に示すように後述するパーミットコードを管理するための暗号化されていない起動処理プログラムと一対にされた配給ソフトとしてソフト供給元から供給される。

【００２９】図示しない入力手段により所望のアプリケーションソフトの実行指令を受けると（ステップ１）、ＣＰＵ２は、先ずスクランブル符号発生器６の出力を停止すると共に、暗号処理器７の復号機能を停止するための指令信号ｓ１を出力する（ステップ２）。そして次のステップ３で、この配給ソフトがメインメモリ２１にロードされ、更に起動処理プログラムがＣＰＵ２に読み込まれて実行される。

【００３０】この暗号化されたアプリケーションソフトEsf２（DapX(+)F(ReX)）がメインメモリにロードされる際に、メインメモリ２１に指定される先頭アドレスＴｏはその管理状況によって変わるが、前記したようにそれに続くアドレスは連続的に指定されるため、この時の先頭アドレスＴｏからの相対位置を示す相対アドレス群は、スクランブル時に用いられた相対アドレス群ReXと同じデータとなる。

【００３１】この配給ソフトの供給手段としてはハード

6

ディスク２２、ＣＤ－ＲＯＭ２３、フロッピーディスク２４及び、モデム２５等による方法が考えられるが、その方法は一般的に行なわれている方法を採用できるため、ここでその詳しい説明は省略する。

【００３２】続くステップ４から１０のステップは、いま実行された起動処理プログラムに従って実行され、次のステップ４では、この暗号化アプリケーションソフト専用の作業鍵Ksf２を公開鍵Kpmpで暗号化したパーミットコードEpmp（Ksf２）がハードディスク２２に既にメモリされているかを確認する。この公開鍵Kpmpは、上記したように図１のＣＭＰ１固有の秘密鍵Ksmpに対応するもので、この秘密鍵Ksmpのみによって復号することが可能となる。

【００３３】従って、この場合のパーミットコードEpmp（Ksf２）は、公開鍵で作業鍵を暗号化したものであり、アプリケーションソフト供給元が、ＣＭＰ１のユーザーから公開鍵Kpmpを入手して作成する。ステップ４でこのパーミットコードの存在が確認できないと、ＣＭＰ１のユーザーは何らかの方法でソフト供給元からこれを入手し（ステップ５）、このアプリケーションソフトの専用ファイルにパーミットコードを入力し、ハードディスク２２にこれを保存する（ステップ６）。この様にパーミットコードの入力保存は、通常は暗号化されたアプリケーションソフトと対になったこの起動処理プログラムによって行なわれる。

【００３４】次に、パーミットコードEpmp（Ksf２）の存在を確認するとこれを読み込み（ステップ７）、内蔵の保管レジスタ１０に保管している秘密鍵Ksmpでこれを復号化して得た作業鍵Ksf２を保管レジスタ８に保管する（ステップ９）。

【００３５】次にＣＰＵ2は、スクランブル符号発生器６の出力を再開し、暗号処理器７の暗号化／復号化機能を起動するための指令信号ｓ１を出力する（ステップ１０）。そしてステップ１１に至り、必要に応じてこのメインメモリ２１に保管された暗号化されたアプリケーションソフトEsf２（DapX(+)F(ReX)）がＣＰＵ２へ読み込まれて実行される。

【００３６】従って、この時この暗号化されたアプリケーションソフトEsf２（DapX(+)F(ReX)）は、暗号処理器７によって作業鍵Ksf２で復号化されて（DapX(+)F(ReX)）となる。また上記のごとく、この時のアドレスデータから生成されるスクランブル符号はF(ReX)であり、加算器１１は、これ等の排他論理和を行なってデスクランブルされたアプリケーションソフトDapXをＣＰＵ２のデータポート１７に出力する。

【００３７】以上のようにして、アドレススクランブルと暗号化されたアプリケーションソフトがＣＰＵ２に読み込まれる。このようにして暗号化されたアプリケーションソフトとパーミットコードの組合せにより、下記の長所をあげることができる。

7

【００３８】長所１．ソフト供給元は、供給するソフトを所定のＣＭＰでのみ利用させることが出来る。ＣＭＰの秘密鍵Ｋｓｍｐがその外部に現れることがなく、ユーザーを含む第３者によってチェック出来ないため。２．暗号化されたアプリケーションソフトをＣＤ－ＲＯＭ等の複製可能な媒体により市場に出すことができる。同種のアプリケーションソフトは、同じ作業鍵で暗号化されるため。３．異種のアプリケーションソフトは、それぞれ専用の作業鍵で暗号化されなければならないが、アプリケーションソフトの種類だけ異なる作業鍵を用意すればよい。

【００３９】次に、ユーザーによって作成され、一旦メインメモリ２１に保存したデータを、暗号化して再度メインメモリ２１に保存しなおす過程を図３のフローチャートを用いて説明する。この場合、このデータを作成する段階では、データの暗号化は行なわれないものとする。

【００４０】このプログラムが起動すると、上記したようにユーザーによって作成されたデータ群ＤｓＸが暗号化されないままメインメモリ２１に保管される（ステップ２１）。次にＣＰＵ２は、キーボード等の入力手段（図示せず）から入力するユーザーからの指示に基づいてこのデータ群ＤｓＸを共通鍵方式で暗号化するための作業鍵Ｋｕｓｒを決定し（ステップ２２）、これを保管レジスタ８に出力して保管する。

【００４１】次にＣＰＵ２は、メインメモリ２１のデータ群ＤｓＸを読み込むために、スクランブル符号発生器６の出力を停止すると共に、暗号処理器７の機能を一時停止するための指令信号ｓ１を出力する（ステップ２５）。この時データ群ＤｓＸはそのままＣＰＵ２に読み込まれるが１バイト毎にアドレス指定されて読み込まれ（ステップ２６）、この段階では最初の１バイト分のデータＤｓ1がＣＰＵ２に読み込まれる。尚、データ群ＤｓＸは、１バイトデータＤｓ1, Ｄｓ2, ……ＤｓＮの集合として標記する。

【００４２】次にＣＰＵ２は、スクランブル符号発生器６の出力を再開し、暗号処理器７の復号化機能を起動するための指令信号ｓ１を出力する（ステップ２７）。この状態でＣＰＵ２は、取り込んだ１バイト分のデータＤｓ1を再度メインメモリ２１に書き込む。

【００４３】この書き込み過程で、先ずこのデータの番地を指定するアドレスに対応してスクランブル符号発生器６から出力されるスクランブル符号Ｆ(Re1)と、このデータＤｓ1との排他論理和（Ｄｓ1(+)Ｆ(Re1)）によるアドレススクランブルが行なわれ、これがデータ経路１４に現れる（ステップ２８）。尚、スクランブル符号Ｆ(Re1)は、上記したデータ群ＤｓＸの各１バイトデータに対応して生成されるスクランブル符号Ｆ(Re1), Ｆ(Re2), ……Ｆ(ReN)の１番目に相当し、これ等の集合をＦ(ReＸ)と標記する。

8

【００４４】更にこのデータ（Ｄｓ1(+)Ｆ(Re1)）は、暗号処理器７によって作業鍵Ｋｕｓｒで暗号化され(ステップ２８)、Ｅｕｓｒ（Ｄｓ1(+)Ｆ(Re1)）の状態でメインメモリ２１に書き込まれる。次に、再びステップ２４に戻って、暗号化するデータ群ＤｓＸが終了するまで１バイトづつ同様の暗号化処理が繰り返される。

【００４５】そして、処理するデータＤｓＸがなくなった段階でステップ３０に至り、スクランブル符号発生器６の出力停止と、暗号処理器７の暗号化機能停止のための指令信号ｓ１を出力する。そしてこのメインメモリ２１に書き込まれた暗号化されたデータを、ＨＤ，ＦＤ等の補助記憶装置に保存して（ステップ３１）このプログラムを終了する。

【００４６】尚、ＣＰＵ２のデータポート１７部にキャッシュメモリ２６を設けると、入出力するデータをためることができ、暗号化処理効率を上げることが出来る。即ち、ステップ２６で複数バイト、例えば１００バイト分のデータを連続して読み込み、ステップ２８でこの１００バイト分のデータを連続してスクランブル及び暗号化することによって、ステップ２５とステップ２７の各機能の切り換え行程をその分省略することが出来る。

【００４７】上記の例では、ユーザーが作成したデータを暗号化する行程を説明したが、前記したアプリケーションソフト提供元が、暗号化プログラムを作成する場合も同様の行程で処理することが出来る。アプリケーションソフトを暗号化する場合には、作成したアプリケーションソフトをデータ群ＤａｐＸとしてメインメモリ２１に一旦保管し、共通鍵方式の秘密鍵として作業鍵Ｋｓｆ２を設定し、上記した図３のフローに従って処理すれば、前記した暗号化されたアプリケーションソフトＥｓｆ２（ＤａｐＸ(+)Ｆ(ReＸ)）を補助記憶装置に保管することが出来る。

【００４８】また上記の実施例の場合、ステップ２で一旦データをメインメモリに作成し、その後暗号化処理を行なうケースを示したが、データを作成する過程で直接暗号化してメモリに出力する方法も考えられる。この場合、作成したデータをメインメモリに出力する際に、その過程で１バイト毎にデータのアドレススクランブルと暗号化を行ない、メインメモリに保存するように構成すればよい。

【００４９】また前記実施例では、オフセットレジスタ４にアドレス群の先頭アドレスをメモリするように説明したが、これに限定されるものではなく、相対アドレスＲｅｉを導き出せる基準値であればよい。

【００５０】次に、ＣＭＰ１内に設けられた秘密メモリ５の働きについて説明する。このため、ソフト供給元によってその稼働時間が管理されたアプリケーションソフトを、例えば図４乃至図７の各フローチャートに従ってＣＤ－ＲＯＭ２３から取り込み、実行する場合を例に説明する。

【００５１】これ等のフローチャートの説明の前に、理解を容易にするために秘密メモリ５の働きについて、その要点を先ず説明する。図９は、秘密メモリ５に保存されるデータの内容を示すが、ここには、ＣＭＰ１が実行するｎ種類のアプリケーションソフトに１対１で対応するパスワードＰｗが保存される。

【００５２】このパスワードＰｗは、秘密メモリ５内で唯一の値であればどのような値でも良く、後述する図４のインストールフローのステップ７５では、生成される乱数によって決定される例を示す。この様にして決定されるパスワードＰｗは、秘密メモリ５内に保存されると共に、インストール実行中のアプリケーションソフトの環境設定ファイルにも保存され、後述する図５の実行フローのステップ４７で秘密値Ｎraの照合を行なう際に利用される。

【００５３】この秘密値Ｎraは、同じく図６の実行フローのステップ５７でアプリケーションソフトの実行が終了する毎に任意に発生される乱数などによって更新され、稼働可能時間を示す残量時間Ｔreと共に実行中のアプリケーションソフトの時間データファイルＤdat（Ｄdat＝Ｔre＋Ｎra'）としてハードディスク２２に保存される（ステップ５９）。更に、ここで更新された秘密値Ｎra'は、ステップ６０で秘密メモリ５内に確保された実行中のアプリケーションソフトに対応するパスワードＰｗによって同メモリ内で識別される秘密値Ｎraに上書きされる。尚、図５、６に示す実行フロー内において、秘密値の更新前後を区別するために、更新前の秘密値をＮraで、また更新後の秘密値をＮra'としてそれぞれを示す。

【００５４】このステップ６０での秘密値の更新に際しては、後述するように安全性を高めるために秘密メモリ５の更新前の秘密値Ｎraによる照合が行なわれる。よって、更新時には（Ｐｗ，Ｎra，Ｎra'）が用意され、パスワードＰｗによって識別された秘密メモリ５内のＮraと照合のために用意したＮraとが一致した場合のみ、新しい秘密値Ｎra'によって書換が行なわれる。このようにしてユーザー等の第３者によるＮraの安易な変更を防止する。

【００５５】以上の様に秘密値Ｎraを秘密メモリ５に保管すると共に、時間データファイルＤdatとしてハードディスク２２内に保存し、このアプリケーションソフトが実行される毎にこれ等を照合することによって、ハードディスク２２に保管されている時間データファイルＤdatが第３者によって改ざんされていないかをチェックすることができる。

【００５６】例えばユーザーが、残量時間Ｔreが十分残っている段階で時間データファイルＤdatをコピーして予め別途確保しておき、このアプリケーションソフトが実行されて残量時間が僅かになった段階で、ハードディスク２２に保存される時間データファイルＤdatと置き換えたとする。しかしながらこの場合、この置き換えられた時間データファイルの秘密値と秘密メモリ５に保存してある秘密値を照合することによって、このデータが正規のデータでないことが判明する。なぜならば、この場合の両者の秘密値Ｎraは、生成されたタイミングが異なるため、当然異なった値となっているからである。

【００５７】以上の記載から理解されるように、秘密メモリ５内のデータは、ユーザーを含め第３者によって操作出来ないようにすることが好ましい。従って、ＣＭＰ１の外部に秘密メモリ５のデータが出力されないように構成される必要がある。

【００５８】図４乃至図７の各フローチャートは、上記したように、不正なデータの改ざんを防ぐべく、秘密メモリ５を使って実施される秘密データ管理方法の全体の手順を示すもので、以下その流れを順に説明する。

【００５９】先ず、所望のアプリケーションソフトを入手してハードディスク２２にインストールし、更に必要な諸環境を初期設定するまでの流れを図４のインストールフローに従って説明する。所望のアプリケーションソフトの入手方法は種々考えられるが、このソフトが入ったＣＤ－ＲＯＭを所定のルートで入手し（ステップ７１）、インストールする例を記述する。尚、このインストールフローは、アプリケーションソフトに付随するセットアッププログラムファイルに基づいて実行されるものである。

【００６０】この場合、このＣＤ－ＲＯＭ２３がＣＤ－ＲＯＭドライブにセットされ、図示しない入力手段によってインストールの指示を受けると、そのアプリケーションソフトがハードディスク２２にインストールされる（ステップ７２）。他のインストール例として、アプリケーションソフトをモデム２５を介してダウンロードし、ハードディスク２２にインストールするようにしてもよい。

【００６１】但し、この時インストールされたアプリケーションソフトＤexeＸは、前記した方法でスクランブル符号Ｆ（ＲeＸ）によるアドレススクランブルと作業鍵Ｋsf３による暗号化が行なわれ、Ｅsf３（ＤexeＸ(+)Ｆ(ＲeＸ)）の状態でハードディスクに保存されているものとする。

【００６２】そして次のステップ７３では、作業鍵Ｋsf３を確保し、保管レジスタ８にこれを保管する。そのため、このステップでは、図２のフローのステップ５からステップ９で説明したように所定の手順でソフト供給元から入手したパーミットコードＥpmp（Ｋsf３）をハードディスク２２に保管し、復号部９で復号化する作業が含まれる。

【００６３】そして次のステップ７４では、スクランブル符号発生器６の出力を開始し、暗号処理器７の暗号化／復号化機能を起動するための指令信号ｓ１を出力する。これにより、後述するステップ８２で両機能がオフ

11

とされるまで、ＣＭＰ１とメインメモリ２１等の外部記憶装置との間で入出力するデータはすべて暗号化されたものとなる。

【００６４】次のステップ７５では、前記したようにこのアプリケーションソフトに１対１で対応するパスワードＰｗ３を生成すべく乱数を発生させ、まだ秘密メモリ５内に存在しない数値であることを確認して確定し、同メモリ内にこれを確保する。そして、インストールされたアプリケーションソフトDexeXの環境設定ファイルAcfg３を作成し、この中に生成したパスワードＰｗ３を書き込んでハードディスク２２に保存する（ステップ７６）。

【００６５】次に、このアプリケーションソフトの稼働可能な残量時間Treをゼロとし（ステップ７７）、乱数を発生させて秘密値Nraを生成する(ステップ７８)。そして残量時間Treと秘密値Nraとからなる時間データファイルDdat（Ddat＝Tre＋Nra）を作成し、ハードディスク２２にこれを保存する（ステップ７９、８０）。

尚、この時作成される時間データファイルDdatは、必要に応じて一旦メインメモリ２１に保存されるが、この際のデータは当然暗号化され、且つスクランブルされたデータとして保存される。

【００６６】また、この時生成した秘密値Nraは、図９に示すように秘密メモリ５内に確保されたパスワードＰｗ３に対応してこれに保存され（ステップ８１）、このパスワードＰｗ３によって照合などの操作が可能とされる。そして、ステップ８２でスクランブル符号発生器６の出力と、暗号処理器７の暗号化／復号化機能と停止してこのフローを終了する。

【００６７】以上のステップによって所望のアプリケーションソフトDexeXが暗号化された状態でハードディスク２２にインストールされ、更に必要な諸環境が初期設定されたことになる。

【００６８】次に、所望のアプリケーションソフトを実行する場合の手順について、図５，６の実行フローチャートを参照しながら説明する。説明を容易にするため、ここで実行するアプリケーションソフトを、上記の説明でインストールされたDexeXとする。

【００６９】ステップ４１でこのアプリケーションソフトが実行されると、このソフトの読み込みが行なわれる（ステップ４２）。このステップ４２の行程は、前記した図２の示すフローチャートのステップ２からステップ１０の行程に対応するもので、既に説明したように暗号化スクランブルされたアプリケーションソフトEsf３（DexeX(+)F(ReX)）がメインメモリ２１にロードされると共に、パーミットコードEpmp（Ksf３）が復号され、作業鍵Ksf３が保管レジスタ８に保管される。

【００７０】更に、スクランブル符号発生器６の出力を開始し、暗号処理器７の暗号化／復号化機能を起動するための指令信号ｓ１を出力する。これにより、後述する

12

ステップ４９で両機能がオフとされるまで、ＣＭＰ１とメインメモリ２１等の外部記憶装置との間で入出力するデータはすべて暗号化されたものとなる。

【００７１】尚、図２で説明したように、この暗号化スクランブルされたアプリケーションソフトEsf３（DexeX(+)F(ReX)）には前記した起動処理プログラムが添付されているものとする。また今回は、図４のフローに基づくインストール時にパーミットコードの保管が行なわれるので、結果的に図２中の起動処理プログラムによるステップ４からステップ６の行程を省略できる。

【００７２】そして、このソフトの環境設定ファイルAcfg３を読み出し、この中にパスワードＰｗがあるか確認する（ステップ４３）。このケースでは、パスワードＰｗ３が存在するのでステップ４４に至るが、もし存在しない場合、このソフトのインストール時にエラーが生じてパスワードが生成されなかったことを意味するので、このことを図示しない表示手段で表示して再インストールの指令を出し（ステップ４５，４６）、更にステップ４９でスクランブル符号発生器６の出力と暗号処理器７の機能とを停止した後、待機状態に戻る。

【００７３】次にステップ４４でこのアプリケーションソフトの時間データファイルDdat（Ddat＝Tre＋Nra）が読み出される。この時、当然作業鍵Ksf３による復号化とアドレスによるデスクランブルが行なわれる。尚、この時読み出される暗号化された時間データファイルDdatは、必要に応じてメインメモリ２１にそのままの形で保管される。

【００７４】次に、秘密メモリ５内の秘密値Nraと時間データファイルDdatの秘密値との照合が行なわれる（ステップ４７）。今は、上記図４のインストールフローの説明でインストールされたアプリケーションソフトDexeXの秘密値が照合されるため、この時間データファイルDdatが何らかの方法で改ざんされないかぎりこれら２つの秘密値は一致し、ステップ５０に移行することが理解される。

【００７５】もしここで、前記したような時間データファイルDdatの置き換えが行なわれていると、当然これら２つの秘密値が異なるためステップ４８に移行し、後述する処理が行なわれる。

【００７６】ステップ５０乃至ステップ５３は、アプリケーションソフトDexeXを実行処理すると同時に残量時間を監視するフローになっている。即ち、ステップ５０では実行時間を計測して逐次残量時間を更新し、ステップ５１では残量時間がゼロになったかを監視し、更にステップ５３ではこの実行の停止指令が入ったかどうかを監視する。

【００７７】また、このステップ５２では、必要に応じて順次メインメモリ２１にロードされたアプリケーションソフトEsf３（DexeX(+)F(ReX)）がＣＰＵ２へ読み込まれるが、その際には前記した復号化、及びデスク

ランブルが逐次実行される。

【００７８】この実行中に残量時間がゼロになるとステップ５１でこれを判定し、一旦ソフト処理を終了した後（ステップ５４）、図示しない表示手段によって例えば「残量時間がありません。」等の表示を行なって（ステップ５５）、後述するようにソフト供給元に残量時間の追加を申請するプログラムに入る（ステップ５６）。今は、インストールされたばかりのアプリケーションソフトDexeXの例を想定して記述しているので残量時間がゼロになっており、当然このルートをたどる。

【００７９】このステップ５６の残量時間追加申請プログラムは、図７に示すフローチャートに示す手順によって行なわれる。即ちユーザーは、残量時間がなくなって暗号化された状態の時間データファイルDdatを何等かの方法でソフト供給元に届ける（ステップ９１）。その方法は種々考えられるが、例えばモデム２５を介して行なわれるインターネットなどのパソコン通信によって行なってもよい。

【００８０】一方、ソフト供給元では、このファイルの復号化とデスクランブルを行なってデータを復元し、このDdat（Ddat＝Tre＋Nra）の残量時間Treの追加補充を行なう（ステップ９２）。その補充量は、ユーザーの依頼に応じて決定されるが、対価の補充料金の支払については別途行なわれるものとし、ここでの説明は省略する。

【００８１】ソフト供給元では、この残量時間Treの補充を行なった後、一緒に送られてきた秘密値と共に、再びアドレススクランブルと作業鍵Ksf３による暗号化を行なった後ユーザーに返信する（ステップ９３）。この方法も、パソコン通信を介して行なってもよいし、フロッピィディスクに納めて郵送するようにしてもよい。

【００８２】ユーザーは、この時間補充された時間データファイルを再度入手し、前のデータファイルと置き換える如くこれをハードディスク２２に保存し（ステップ９４）、この残量時間追加申請のフローを終了する。従って、この時のDdat（Ddat＝Tre＋Nra）において、残量時間Treは所定量補充されているものの、秘密値Nraは、秘密メモリ５に保存されている秘密値と同じままであることが理解される。

【００８３】ユーザーは、このアプリケーションソフトDexeXを引き続いて実行したい場合、再度図５の実行フローのステップ４１に戻らなければならない。然し乍ら今回は、残量時間が所定量補充されているので、ステップ４１で実行が指令されると、ステップ５０乃至ステップ５３のフローが繰り返されて、アプリケーションソフトDexeXの実行状態が継続されることになる。

【００８４】そして、残量時間Treがゼロとなる前に、ユーザーによるソフト処理終了の指令を受けると、次のステップ５７に至る。このステップ５７では、乱数を発生して秘密値Nra'を更新し、その後この更新した秘密値Nra'とこの時のソフトの実行時間が差し引かれた残量時間Treとを対にして新たな時間データファイルDdat（Ddat＝Tre＋Nra'）を作成する（ステップ５８）。尚、この時作成されたデータDdatは、必要に応じて一旦メインメモリ２１に保存されるが、この際のデータは当然暗号化され、スクランブルされたデータとして保存される。

【００８５】そしてこれをハードディスク２２に保存されている更新前の時間データファイルに代えて保存する（ステップ５９）と共に、更新された秘密値Nra'を、秘密メモリ５内に確保されたパスワードPw３によって識別可能に同メモリ内に保持されている更新前の秘密値に代えて保存する（ステップ６０）。

【００８６】この時、前記した更新前の秘密値Nraの照合が行なわれる。図１１に示すフローチャートは、ステップ６０内で行なわれるこの照合動作の流れを示すもので、ステップ６０－１では秘密メモリ５内の秘密値Nraと照合のために別途用意されたNraとが合致するかをチェックする。これ等が合致する場合はステップ６０－２で上記したデータの更新が行なわれ、合致しない場合は、ステップ６０－３に至って、図示しない表示手段によって、「不正が行なわれました。」等の表示を行ない、図５のBへ戻るように構成しても良い。

【００８７】そして再度このアプリケーションソフトDexe３が実行されると、上記した各ステップを経てステップ４７に至り、ここで秘密値が照合される。この時、秘密メモリから読み出した秘密値とハードディスク２２から読み出された秘密値とは一致するはずであるが、もし、ユーザーによって前記したようなデータの改ざんがあるとこれ等は一致しない。この時はステップ４８に至って「不正行為が行なわれました。」等のメッセージを表示し、実行を中止すべくステップ４９を経由して待機状態に戻る。

【００８８】一方、このアプリケーションソフトDexeXの使用を中止し、アンインストールする場合には、このソフトに設定されたパスワードPw３と秘密値Nraを秘密メモリ５から消去する必要がある。これを行なわないと同メモリ内に不要な秘密値が蓄積されることになる。またこの消去作業はアンインストール時のみに、またハードディスク２２にパスワードが存在する段階で行なわれる必要があるため、例えば、図８に示すフローに従って行なわれる。

【００８９】即ち、図示しない入力手段によってアンインストールの指示を受けると、ステップ１０１に至り、先ずパスワードと秘密値の消去が行なわれる。この際にも、安全性を高めるために前記したような秘密値Nraによる照合が行なわれる。このために（Pw３，Nra）が用意され、パスワードPw３によって識別された秘密メモリ５内のNraと照合のために用意したNraとが一致した場合にのみこのアプリケーションソフトDexeXに対

応するパスワードＰｗ３と秘密値Ｎraとが同時に消去される。そしてこの消去が終了した後、アプリケーションソフトDexeX自体のアンインストールが実行される（ステップ１０２）。

【００９０】次に、本発明の他の実施例について説明する。図１４は、秘密メモリ部５１がＣＭＰ５０の外にあって独立して存在する場合を示し、その他の前記した図１と同じ機能を有する構成要素については、同符号を付してその説明を省略する。図１に対する図１４の構成上の違いは、ＣＭＰ１内部にあった秘密メモリがその外部の秘密メモリ部５１に照合手段５２と共に独立して設けられた点であり、全体的な機能は全く同じである。

【００９１】一方、図１に対する図１４の部分的な機能の相違点は、図１の構成では秘密メモリ５の内部データがＣＭＰ１の外部に出力されないように構成されているのに対し、図１４の構成では秘密メモリ部５１の内部データがその外部に出力されないように構成される点である。

【００９２】従って、図１の構成のＣＭＰ１内に設けられた秘密メモリの働きについて、図４乃至図７のフローチャートを参照して説明したが、図１４構成のＣＭＰ５０外に設けられた秘密メモリ部５１についても、これ等のフローチャートに従って同様の作業が実行できる。しかしながら、図５のステップ４７、図１１のステップ６１－１、及び図８のステップ１０１の各ステップで行なわれる照合作業は、秘密メモリ部５１内部の照合手段５２で行なわれ、YES, NOの結果のみが出力されるように構成されている。

【００９３】次に、本発明の他の実施例について説明する。図１２は、本発明の他の実施例を示す構成図で、前記した図１と同じ機能を有する構成要素については、同符号を付してその説明を省略し、図１の構成及び動作と異なる部分について重点的に説明する。

【００９４】この実施例の場合、暗号化されアドレススクランブルされたデータＤを前記した標記方法に従って標記すると、Escri（D）となる(i=1, 2, 3……N)。但しこの秘密鍵Kscriは、Kscri＝Ksf(+)Ｆ(Rei)ということになる。即ち、このデータＤに用意される作業鍵Ksfと逐次生成されるスクランブル符号Ｆ(Rei)との排他論理和符号によってデータＤを暗号化したものとなる。

【００９５】このようにして暗号化されたアプリケーションソフトを図１２のＣＭＰによって復号化する過程を説明する。いま、このアプリケーションソフトの１バイト単位のデータ群Ｄ1，Ｄ2……ＤＮの集合をＤＸと記述し、これに対応して生成されるスクランブル符号群Ｆ(Re1)，Ｆ(Re2)……Ｆ(ReN)の集合をＦ(ReX)と記述し、この暗号化されたアプリケーションソフトをＥscr（ＤＸ）と記述する。但し、Kscr＝Ksf(+)Ｆ(ReX)である。

【００９６】従って、いま最初の１バイトのデータＤ1を復号する過程を記述すると、この時データバス１２には、暗号化されたデータEscr1（D1）が現れ、スクランブル符号発生器６からはＦ(Re1)が出力される。従ってこの時の暗号処理器７は、作業鍵Ksfとスクランブル符号Ｆ(Re1)の排他論理和符号Ksf(+)Ｆ(Re1)によってデータEscr1.(D1)を復号することになるため、暗号化した時と同じ鍵でこれを復号化して得たデータD1をＣＰＵ２に出力することが出来る。以下同様にして、全ての暗号化されたデータ群を順次復号化してＣＰＵ２に送る。

【００９７】先に、図１に示す構成のＣＭＰ１による、アプリケーションソフトの入手、及びその実行に際してのＣＰＵ２内への読み込み過程について、図２のフローチャートを参照しながら説明したが、図１２構成のＣＭＰ３０もこのフローチャートに従って同様の作業が実行できる。但し、ステップ１１の復号方法が上記した説明に基づいて実行されるところが異なる。

【００９８】また、ユーザーによって作成され、一旦メインメモリ２１に保存したデータを、図１の構成のＣＭＰによって暗号化して再度メインメモリに保存しなおす過程を図３のフローチャートを参照して説明したが、図１２構成のＣＭＰ３０もこのフローチャートに従って同様の作業が実行できる。但し、ステップ２８の暗号化方法が異なり、排他論理和符号（Kusr(+)Ｆ(ReX)）によってデータ群ＤＸを逐次暗号化するものとなる。

【００９９】更に、図１の構成のＣＭＰ１内に設けられた秘密メモリの働きについて、図４乃至図７のフローチャートを参照して説明したが、図１２構成のＣＭＰ３０内に設けられた秘密メモリ５についても、これ等のフローチャートに従って同様の作業が実行できる。但し、フローで実行されるデータの暗号化又は復号化の方法が図１のＣＭＰ１と異り、上記したように排他論理和符号（Kusr(+)Ｆ(ReX)）を作業鍵として行なわれる。

【０１００】次に、本発明の他の実施例について説明する。図１３は、本発明の他の実施例を示す構成図で、前記した図１と同じ機能を有する構成要素については、同符号を付してその説明を省略し、図１の構成及び動作と異なる部分について重点的に説明する。

【０１０１】この実施例の場合、暗号化されアドレススクランブルされたデータＤを前記した標記方法に従って標記すると、Esss（D(+)Ｆ(Re)）となる。但しこの作業鍵Ksssは、Ksss＝Essp(Kusr)ということになる。即ち暗号部４１は、ＣＰＵ２からユーザーによって設定される作業鍵Kusrを入力し、保管レジスタ１０が保管しているこのＣＭＰ４０に固有の秘密鍵Ksspでこれを暗号化した前記作業鍵Ksssを生成し、これを保管レジスタ８に出力する。

【０１０２】暗号処理部７は、この保管レジスタ８に保管された作業鍵Ksssでアドレススクランブルされたデータ（D(+)Ｆ(Re)）を暗号化し、この暗号化したデー

17

タＥsss（Ｄ(+)Ｆ(Re)）をデータバス１２を介してメインメモリ２１等の外部メモリに出力する。

【０１０３】このようにして暗号化されたデータを図１３のＣＭＰによって復号化する過程を説明する。いま、データの１バイト単位のデータ群Ｄ1，Ｄ2……ＤＮの集合をＤＸと記述し、これに対応して生成されるスクランブル符号群Ｆ（Re1），Ｆ（Re2）……Ｆ（ReN）の集合をＦ（ReＸ）と記述し、このスクランブル暗号化されたデータをＥsss（ＤＸ(+)Ｆ(ReＸ)）と記述する。但し、Ｋsss＝Ｅssp(Kusr)である。

【０１０４】従って、いま最初の１バイトのデータＤ1を復号する過程を記述すると、この時データバス１２には、暗号化されたデータＥsss（Ｄ1(+)Ｆ(Re1)）が現れ、スクランブル符号発生器６からはＦ(Re1)が出力される。従って暗号処理器７は、この時作業鍵Ｋsssでデータ Ｅsss（Ｄ1(+)Ｆ(Re1)）を復号化した（Ｄ1(+)Ｆ(Re1)）を加算器１１に出力し、加算器１１はＦ(Re1)でこれをデスクランブルして得たデータＤ1をＣＰＵ２に出力することが出来る。以下同様にして、全ての暗号化されたデータ群を順次復号化してＣＰＵ２に送る。

【０１０５】また、ユーザーによって作成され、一旦メインメモリ２１に保存したデータを、図１の構成のＣＭＰによって暗号化して再度メインメモリに保存しなおす過程を図３のフローチャートを参照して説明したが、図１３構成のＣＭＰもこのフローチャートに従って同様の作業が実行できる。但し、ステップ２３では作業鍵Ｋsss＝Ｅssp(Kusr)を保管レジスタ８に保管する必要があり、このため前記した暗号部４１よる作業鍵Ｋusrの暗号化が行なわれる。

【０１０６】従って、この図１３に構成されたＣＭＰ４０によって暗号化されたデータは、この暗号化したＣＭＰのみによって復号化されることが理解される。更に、作業鍵Ｋsssが二重鍵の構成を有するため、暗号化するデータ群ごとに作業鍵を変えることができるために鍵自体が解読されにくくなり、データの氾濫を防ぐと共にその機密性を高めることが出来る。

【０１０７】次に、本発明の他の実施例について説明する。図１５は、本発明の他の実施例を示す構成図で、前記した図１と同じ機能を有する構成要素については、同符号を付してその説明を省略し、図１の構成及び動作と異なる部分について重点的に説明する。

【０１０８】この実施例の場合、作業鍵の保管レジスタ８を複数Ｐ個有する保管レジスタ部６１を設け、これ等の各保管レジスタに異なる作業鍵Ｋsfi(i=1, 2, ……P)をそれぞれ保管可能とする。この保管レジスタ部６１は、復号部９から出力される作業鍵を保管するレジスタの選択、及び暗号処理部７に出力する作業鍵の選択を、制御ユニット３から出力される鍵選択信号ｓ５によって実行できるように構成されている。

【０１０９】また、オフセットレジスタ部６２には、前記したデータ群の先頭アドレスを保管するための複数Ｐ個のオフセットレジスタ４が、異なる先頭アドレスＴoi(i=1, 2, ……P)をそれぞれ保管可能に設けられ、制御ユニット３から出力される先頭アドレス選択信号ｓ６によって、先頭アドレスＴoiの保存、及び読み出しの対象となるレジスタが選択可能に構成されている。

18

【０１１０】次にこのような構成のＣＭＰ６０の動作について説明する。今、前記した図２の実行準備フローが実行され、そのステップ３の段階でメインメモリ２１にロードされたソフトが図１７に示すようにＭ個のプログラムから構成され、それぞれが異なる作業鍵Ｋsfｊ（j=1, 2, ……Ｍ、Ｍ≦Ｐ）で暗号化されているものとする。

【０１１１】また各プログラムは、前記したようにそれぞれ所定単位のデータの集合であるデータ群で構成され、各データ群の先頭アドレスをＴoｊ（j=1, 2, ……Ｍ）と記す。　この場合の各作業鍵Ｋsfｊは、各プログラムを管理するソフト供給元が前記したパーミットコードとして供給するため、図２の実行準備フローのステップ４からステップ９までのステップが図１のＣＭＰ１と異なる。

【０１１２】即ち、ステップ４からステップ６までで、パーミットコードの存在の確認、入手、保存の各行程を実施するが、今このパーミットコードがＭ個必要となるため、この各行程もＭ回繰り返すように構成される。更に、ステップ７からステップ９までの行程で、パーミットコードを読み込み、それを復号し、生成した作業鍵をレジスタに保存するが、ここでもこの各行程をＭ回繰り返すように構成される。

【０１１３】この時逐次生成される作業鍵Ｋsfｊが、ＣＰＵから出力される指令信号ｓ１に基づいて制御ユニット３から出力される鍵選択信号ｓ５によって選択された１からＭまでの各保管レジスタ８にそれぞれ保管される。

【０１１４】更にステップ１１で、必要に応じてメインメモリ２１に保管された各プログラムがＣＰＵ２に読み込まれて実行される際には、それぞれのプログラムの作業鍵Ｋsfｊを保管するレジスタが選択され読み出され、暗号処理器７に供給される。これと共に、読み出されるプログラムに対応する先頭アドレスがＣＰＵから出力される指令信号ｓ１に基づいて制御ユニット３から出力されるアドレス選択信号ｓ６によって選択される。そして選択された先頭アドレスＴojがスクランブル符号発生器６に出力され、前記したアドレススクランブルが各プログラム毎に実行される。尚、ｉとｊとが対応して管理されることにより、各プログラムとこれを実行する際に読み出される各作業鍵、及び各先頭アドレスとの対応関係が維持される。

【０１１５】次に、本発明の他の実施例について説明する。図１６は、本発明の他の実施例を示す構成図で、前記した図１と同じ機能を有する構成要素については、同

19

符号を付してその説明を省略し、図１の構成及び動作と異なる部分について重点的に説明する。

【０１１６】この実施例の場合、復号部７２と保管レジスタ７３とからなる多重鍵生成部７１が追加構成される。そして保管レジスタ７３は、このＣＭＰ７０に固有の秘密鍵Ｋsmppを保管し、復号部７２は前記した公開鍵Ｋpmpに対応する秘密鍵Ｋsmpを、秘密鍵Ｋsmppに対応する公開鍵Ｋpmppで暗号化したＥpmpp（Ｋsmp）を得て秘密鍵Ｋsmppで復号化し、生成した秘密鍵Ｋsmpを保管レジスタ１０に出力する。

【０１１７】このように構成することにより、ＣＭＰ７０にとって、一対の公開鍵Ｋpmpと秘密鍵Ｋsmpとがすでに登録された固有のものではなく、例えばＣＭＰ７０の供給元から別途これ等の鍵を入手して設定することが出来るなど、種々の態様を可能とするものとなる。

【０１１８】

【発明の効果】本発明によれば、信号処理手段から秘密データを出力することなく、別途用意した参照データが所定の秘密データと同一のものか判断でき、また秘密データの更新又は消去が、その秘密データを知らない者によって実行できないため、実行時間が管理されるアプリケーションソフトの残量時間などの機密情報を、この秘密データと一緒に外部メモリに保存した場合、この機密情報が不正に改ざんされたか否かを高い信頼性の下にチェックすることが出来る。

【０１１９】また、パスワード等の識別コードをこの秘密データに対応して秘密メモリに記憶することにより、この識別コードが秘密データのアドレスの代用とるなため、ユーザー等の第３者による安易な、また不正なデータの変更を防止できる。

【図面の簡単な説明】

【図１】本発明の一実施例を示す構成図

【図２】本発明の動作説明に供するフローチャート

【図３】本発明の動作説明に供するフローチャート

【図４】本発明の動作説明に供するフローチャート

【図５】本発明の動作説明に供するフローチャート

【図６】本発明の動作説明に供するフローチャート

【図７】本発明の動作説明に供するフローチャート

【図８】本発明の動作説明に供するフローチャート

20

【図９】本発明の動作説明に供する秘密メモリの説明図

【図１０】本発明の動作説明に供するアプリケーションソフトの構成図

【図１１】本発明の動作説明に供するフローチャート

【図１２】本発明の他の一実施例を示す構成図

【図１３】本発明の他の一実施例を示す構成図

【図１４】本発明の他の一実施例を示す構成図

【図１５】本発明の他の一実施例を示す構成図

【図１６】本発明の他の一実施例を示す構成図

【図１７】本発明の動作説明に供するメインメモリの説明図

【符号の説明】

１　ＣＭＰ
２　ＣＰＵ
３　制御ユニット
４　オフセットレジスタ
５　秘密メモリ
６　スクランブル符号発生器
７　暗号処理器
８　保管レジスタ
９　復号部
１０　保管レジスタ
１１　加算器
２１　メインメモリ
２２　ハードディスク
２３　ＣＤ－ＲＯＭ
２４　フロッピディスク
２５　モデム
２６　キャッシュメモリ
３０　ＣＭＰ
４０　ＣＭＰ
４１　暗号部
５０　ＣＭＰ
５１　秘密メモリ部
５２　照合手段
６０　ＣＭＰ
６１　保管レジスタ部
７０　ＣＭＰ
７１　多重鍵生成部

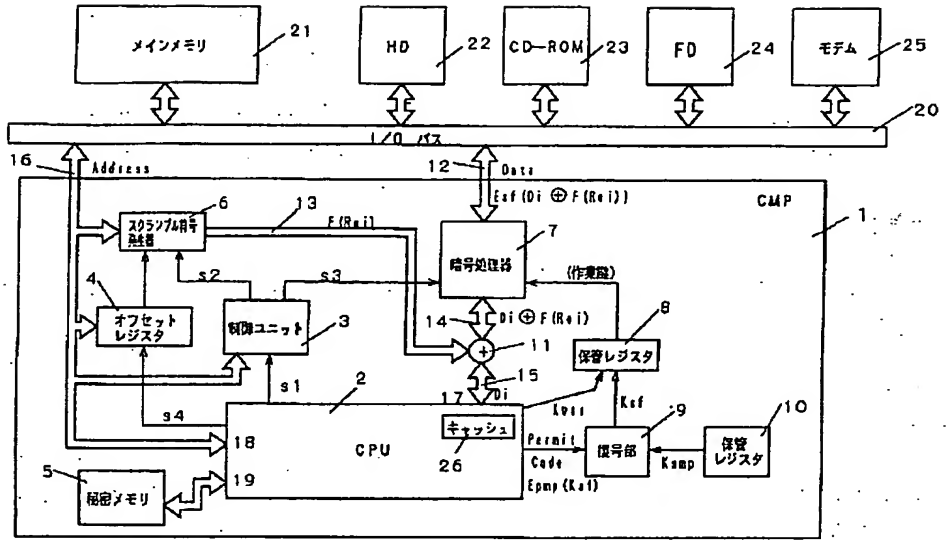
【図９】

パスワード	秘密鍵
PW1	Nrs1
PW2	Nrs1
PW3	Nrs3
PWn	Nrsn

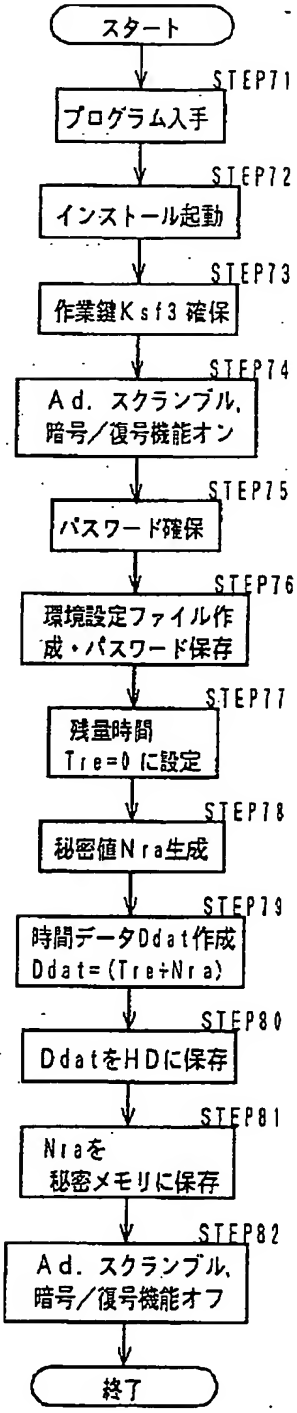
【図１０】

起動処理プログラム	暗号化アプリケーションソフト E=12(D=pX④F(R=X))

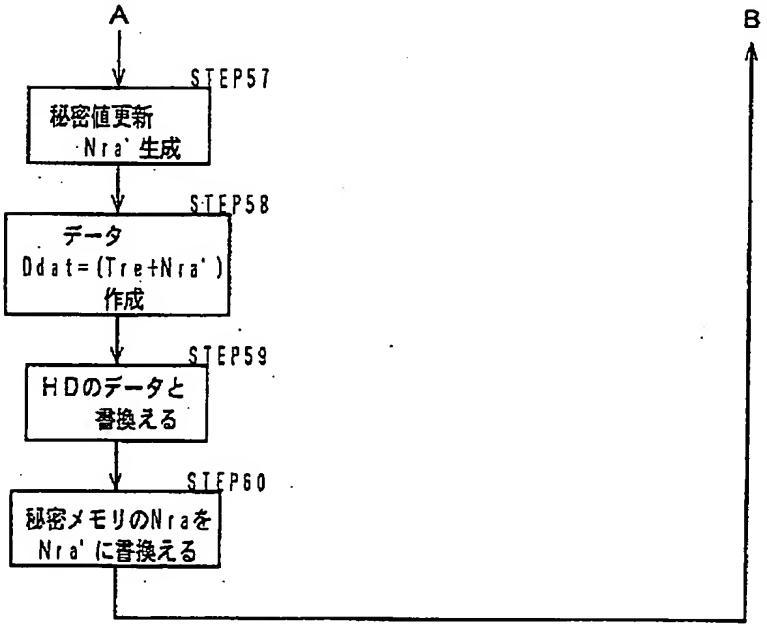
【図１】


メインメモリ
21
HD
22
CD-ROM
23
FD
24
モデム
25
I/O バス
20
16
Address
12
Data
Esf (Di ⊕ F(Rei))
CMP
1
スクランブル符号発生器
6
13
F(Rei)
暗号処理器
7
(作業鍵)
4
オフセットレジスタ
制御ユニット
3
s2
s3
14
Di ⊕ F(Rei)
8
保管レジスタ
11
15
Di
s1
2
17
Ipas
Ksf
9
10
s4
キャッシュ
18
CPU
26
Permit
Code
Epmp (Ksf)
復号部
保管レジスタ
Kmp
5
秘密メモリ
19

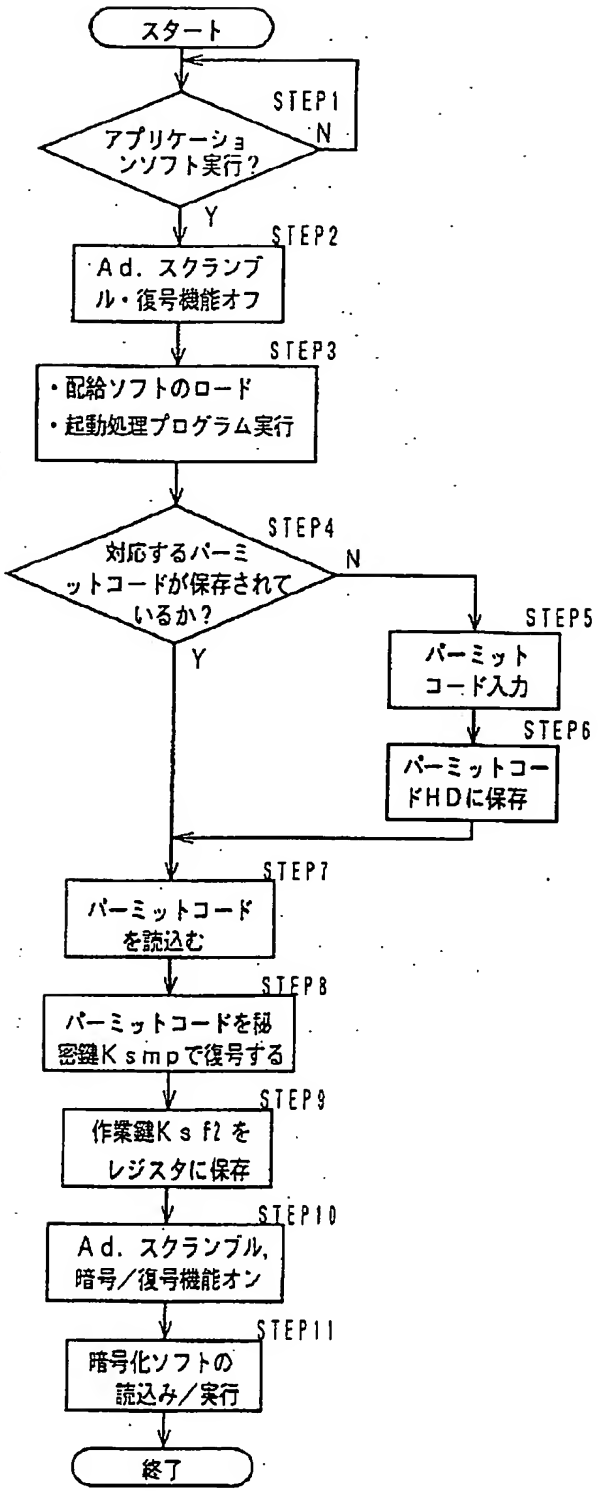
【図４】


スタート
STEP71
プログラム入手
STEP72
インストール起動
STEP73
作業鍵Ksf3 確保
STEP74
Ad. スクランブル,
暗号／復号機能オン
STEP75
パスワード確保
STEP76
環境設定ファイル作
成・パスワード保存
STEP77
残量時間
Tre=0 に設定
STEP78
秘密値Nra生成
STEP79
時間データDdat作成
Ddat=(Tre+Nra)
STEP80
DdatをHDに保存
STEP81
Nraを
秘密メモリに保存
STEP82
Ad. スクランブル,
暗号／復号機能オフ
終了

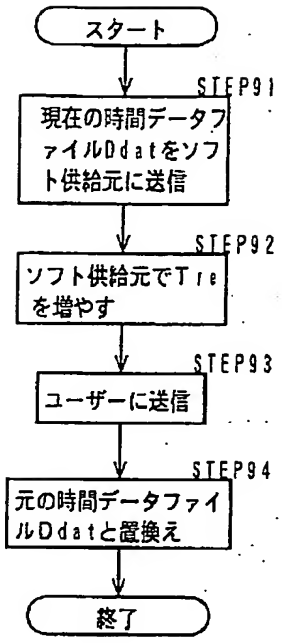
【図６】


A
STEP57
秘密値更新
Nra' 生成
STEP58
データ
Ddat=(Tre+Nra')
作成
STEP59
HDのデータと
書換える
STEP60
秘密メモリのNraを
Nra' に書換える
B

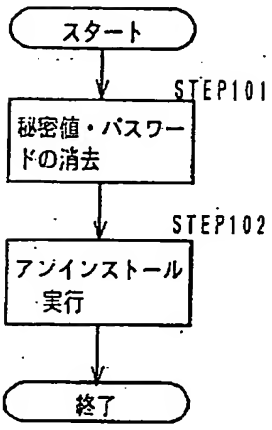
【図２】


スタート
STEP1
アプリケーションソフト実行？
N
Y
STEP2
Ad．スクランブル・復号機能オフ
STEP3
・配給ソフトのロード
・起動処理プログラム実行
STEP4
対応するパーミットコードが保存されているか？
N
Y
STEP5
パーミットコード入力
STEP6
パーミットコードHDに保存
STEP7
パーミットコードを読込む
STEP8
パーミットコードを秘密鍵Ksmpで復号する
STEP9
作業鍵Ksf2をレジスタに保存
STEP10
Ad．スクランブル，暗号／復号機能オン
STEP11
暗号化ソフトの読込み／実行
終了

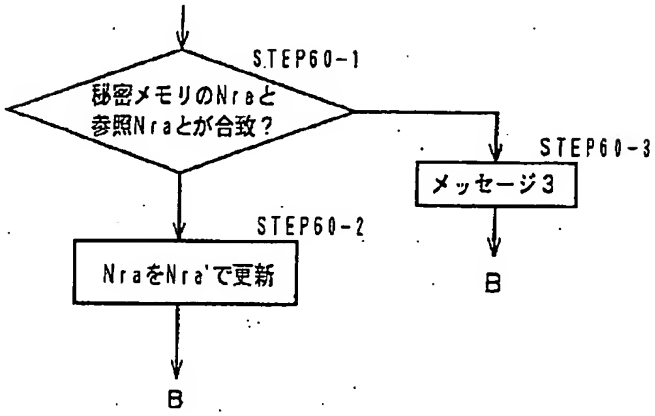
【図７】


スタート
STEP91
現在の時間データファイルDdatをソフト供給元に送信
STEP92
ソフト供給元でTreを増やす
STEP93
ユーザーに送信
STEP94
元の時間データファイルDdatと置換え
終了

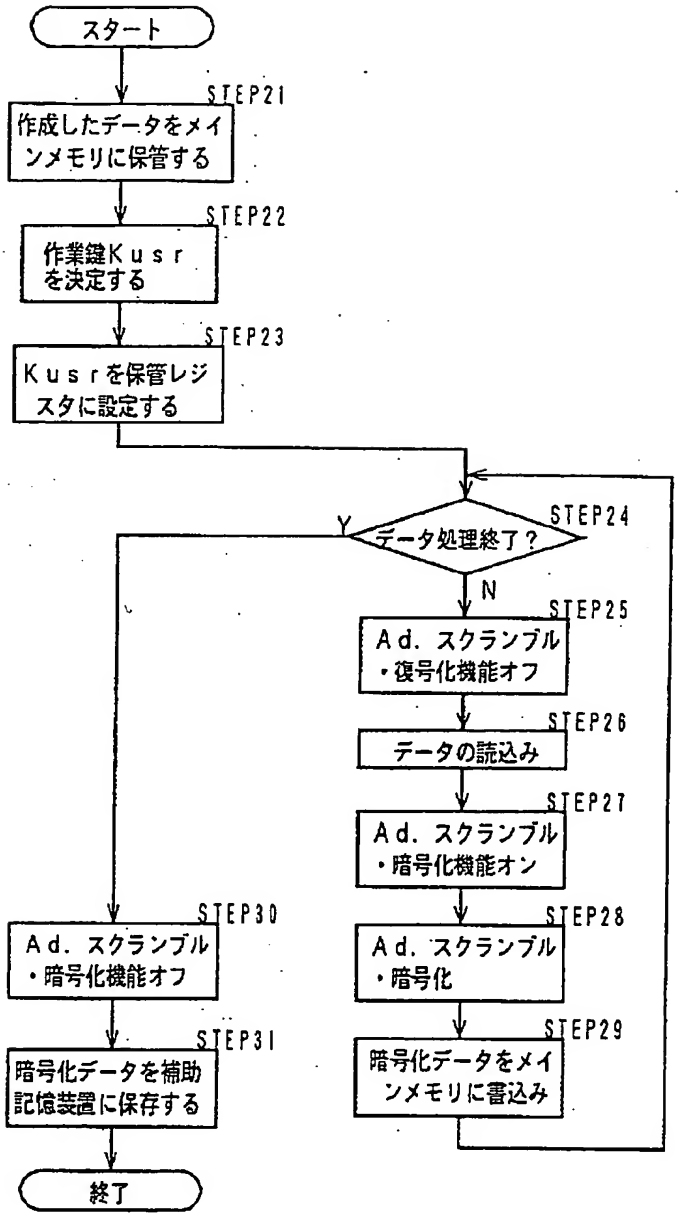
【図８】


スタート
STEP101
秘密値・パスワードの消去
STEP102
アンインストール実行
終了

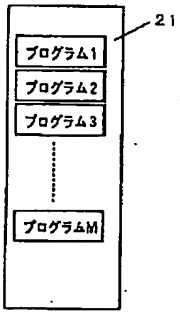
【図１１】


STEP60-1
秘密メモリのNraと参照Nraとが合致？
STEP60-3
メッセージ3
B
STEP60-2
NraをNra'で更新
B

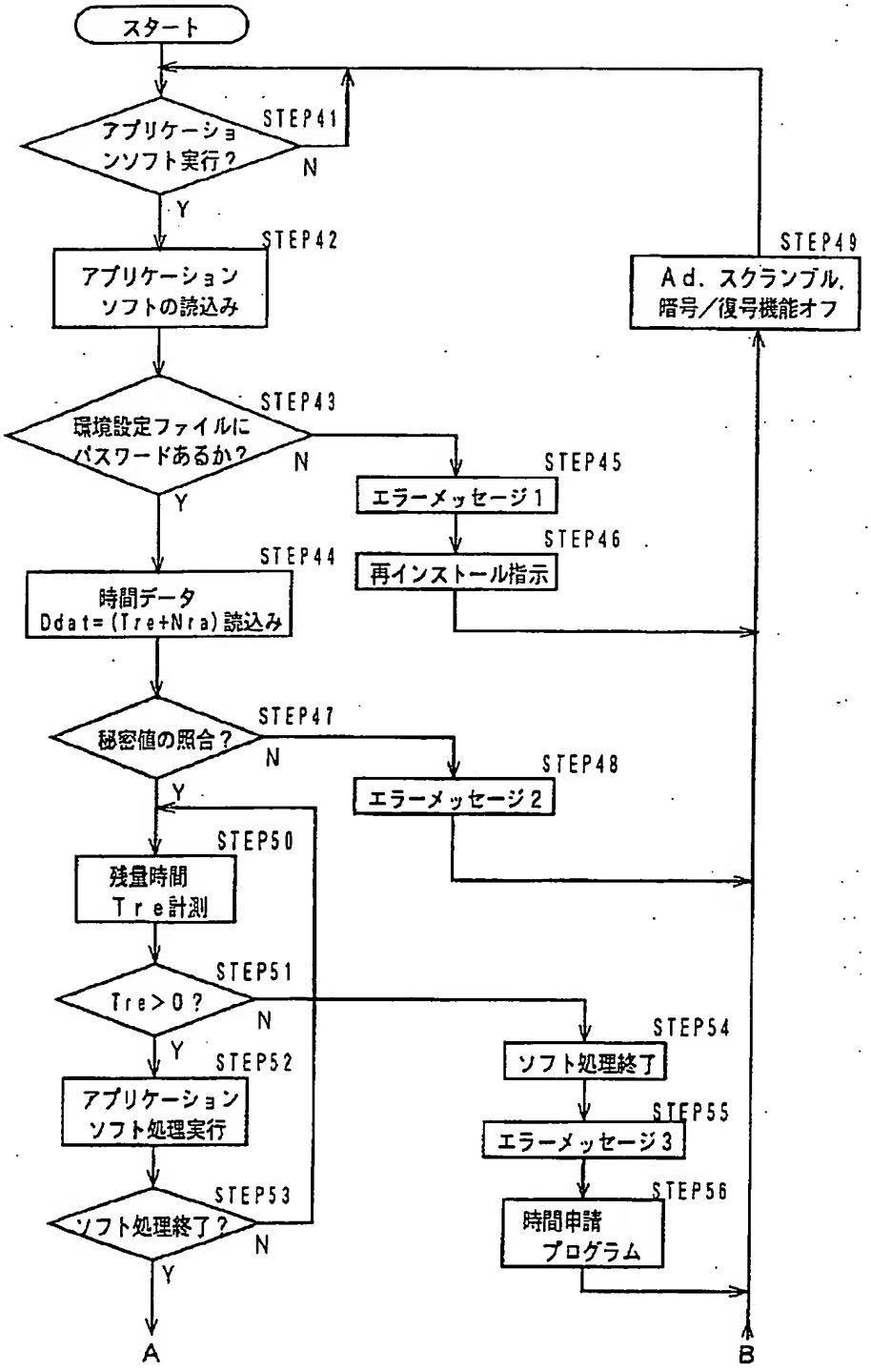
【図３】


スタート
STEP21
作成したデータをメインメモリに保管する
STEP22
作業鍵Ｋｕｓｒを決定する
STEP23
Ｋｕｓｒを保管レジスタに設定する
STEP24
データ処理終了？
Y
N
STEP25
Ａｄ．スクランブル・復号化機能オフ
STEP26
データの読込み
STEP27
Ａｄ．スクランブル・暗号化機能オン
STEP28
Ａｄ．スクランブル・暗号化
STEP29
暗号化データをメインメモリに書込み
STEP30
Ａｄ．スクランブル・暗号化機能オフ
STEP31
暗号化データを補助記憶装置に保存する
終了

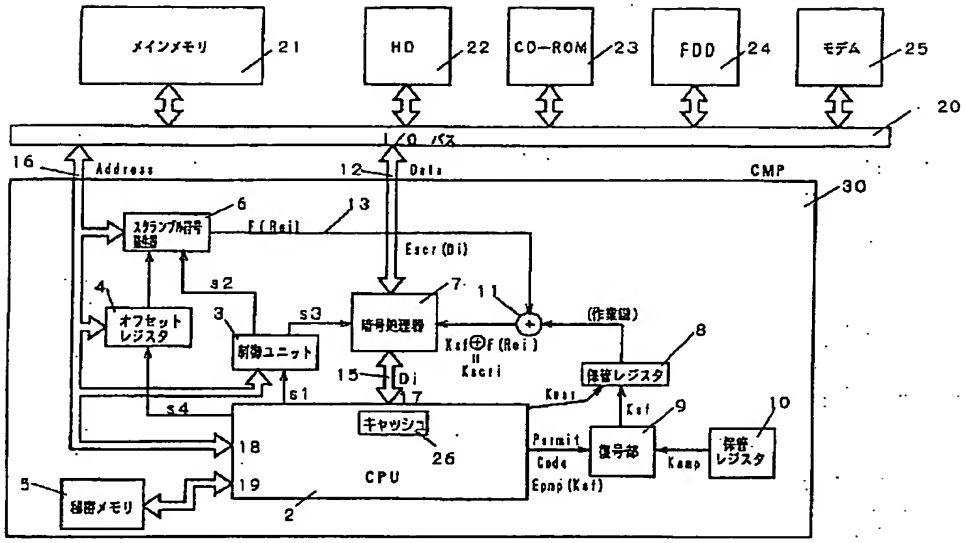
【図１７】


21
プログラム1
プログラム2
プログラム3
プログラムM

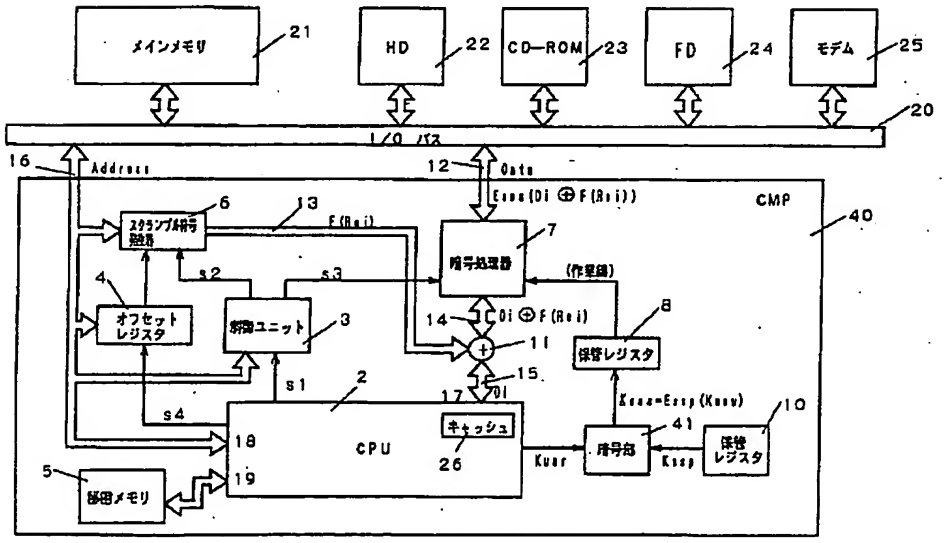
【図５】


スタート
アプリケーションソフト実行？
STEP41
N
Y
アプリケーションソフトの読込み
STEP42
環境設定ファイルにパスワードあるか？
STEP43
N
Y
エラーメッセージ１
STEP45
再インストール指示
STEP46
時間データ Ddat=(Tre+Nra) 読込み
STEP44
秘密値の照合？
STEP47
N
Y
エラーメッセージ２
STEP48
残量時間 Ｔｒｅ計測
STEP50
Tre>0？
STEP51
N
Y
アプリケーションソフト処理実行
STEP52
ソフト処理終了？
STEP53
N
Y
A
ソフト処理終了
STEP54
エラーメッセージ３
STEP55
時間申請プログラム
STEP56
Ａｄ．スクランブル，暗号／復号機能オフ
STEP49
B

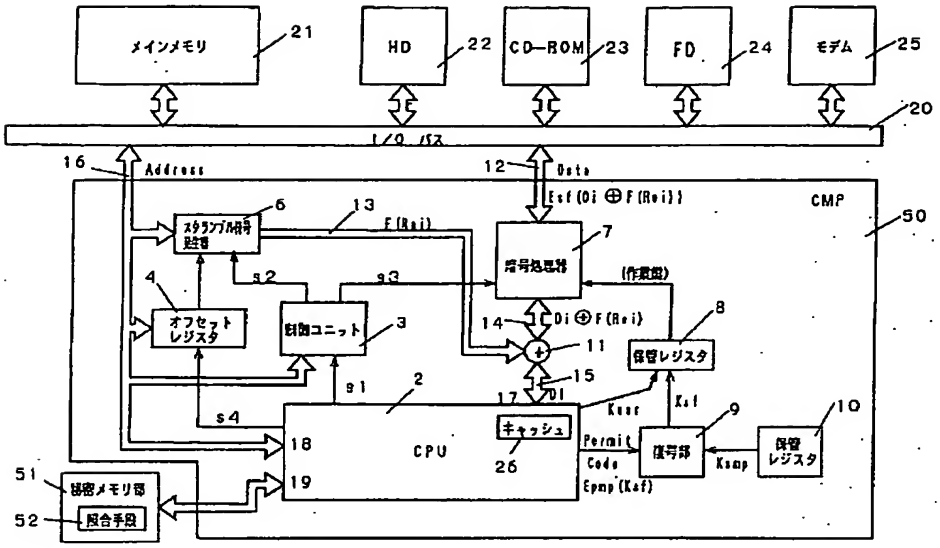
【図１２】


メインメモリ
21
HD
22
CD-ROM
23
FDD
24
モデム
25
20
I/O バス
16
Address
12
Data
CMP
30
6
F(Rai)
13
Escr(Di)
4
オフセット レジスタ
s2
3
制御ユニット
s3
7
暗号処理器
11
(作業鍵)
Xsf⊕F(Roi) = Kscri
8
保管レジスタ
15
Di
7
s1
Ksaj
Ksf
9
10
s4
キャッシュ
18
26
Permit
Code
Epnp(Ksf)
復号部
Kanp
保管 レジスタ
CPU
5
秘密メモリ
19
2

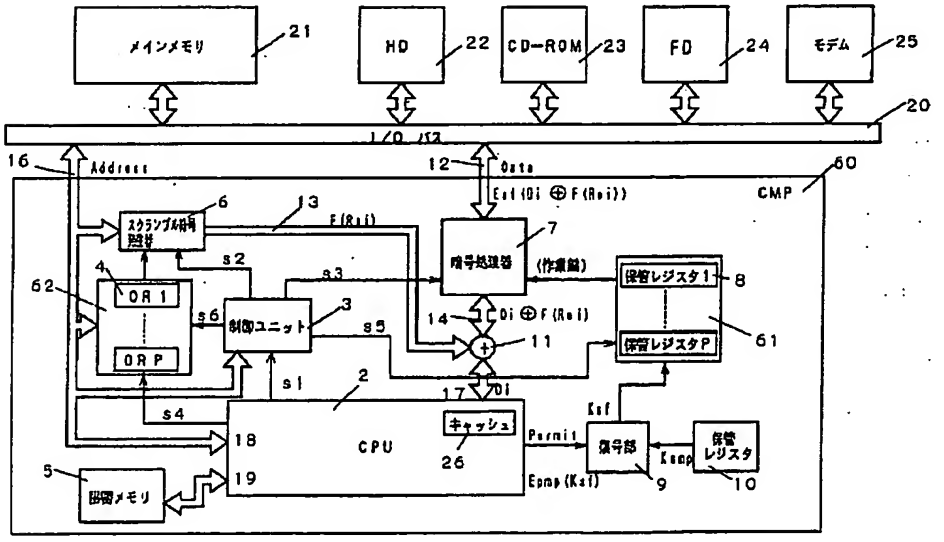
【図１３】


メインメモリ
21
HD
22
CD-ROM
23
FD
24
モデム
25
20
I/O バス
16
Address
12
Data
Esaa(Di ⊕ F(Roi))
CMP
40
6
F(Rai)
13
7
暗号処理器
(作業鍵)
4
オフセット レジスタ
s2
s3
制御ユニット
3
14
Di ⊕ F(Roi)
8
保管レジスタ
11
15
2
s1
17
Di
Ksaa=Esaj(Ksav)
10
s4
キャッシュ
41
18
CPU
26
Ksar
暗号部
保管 レジスタ
Ksaj
5
秘密メモリ
19

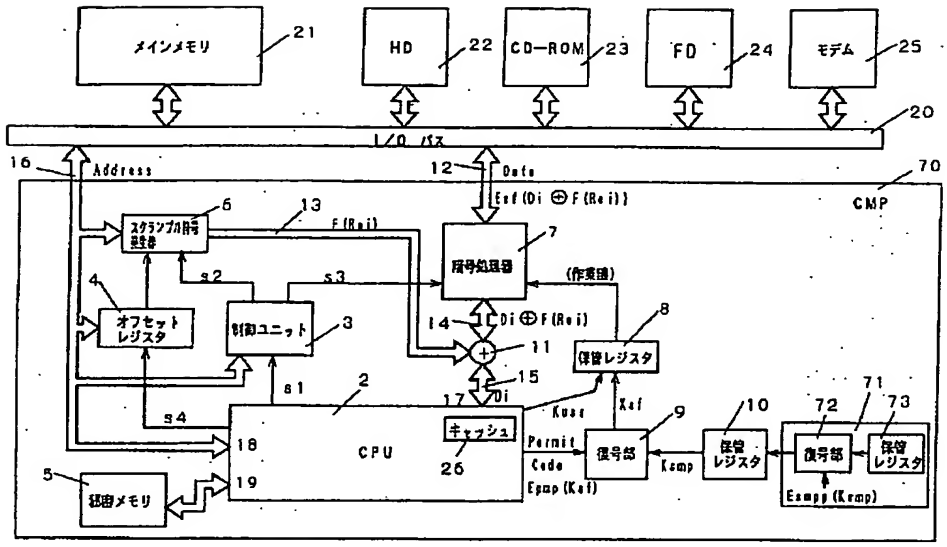
【図１４】


メインメモリ
21
HD
22
CD-ROM
23
FD
24
モデム
25
20
I/O バス
16
Address
12
Data
Enf(Di ⊕ F(Roi))
CMP
50
スクランブル符号発生器
6
13
F(Roi)
7
暗号処理器
(作業鍵)
4
オフセットレジスタ
s2
s3
制御ユニット
3
14
Di ⊕ F(Roi)
8
保管レジスタ
11
15
17
Di
s1
2
Kuor
Kaf
s4
キャッシュ
9
10
18
CPU
Permit
復号部
保管レジスタ
26
Code
Kmp
Epmp(Kaf)
19
51
秘密メモリ部
52
照合手段

【図１５】


メインメモリ
21
HD
22
CD-ROM
23
FD
24
モデム
25
20
I/O バス
16
Address
12
Data
Enf(Di ⊕ F(Roi))
CMP
60
スクランブル符号発生器
6
13
F(Roi)
7
暗号処理器
(作業鍵)
保管レジスタ1
8
4
62
OR 1
OR P
s6
制御ユニット
3
s2
s3
s5
14
Di ⊕ F(Roi)
11
保管レジスタP
61
17
Di
s1
2
s4
Kaf
キャッシュ
18
CPU
Permit
復号部
保管レジスタ
26
Kmp
19
Epmp(Kaf)
9
10
5
秘密メモリ

【図１６】


メインメモリ
21
HD
22
CD-ROM
23
FD
24
モデム
25
I/O バス
20
16
Address
12
Data
70
CMP
Enf(Di ⊕ F(Rei))
6
13
F(Rei)
7
暗号処理器
(作業領域)
4
s2
s3
オフセットレジスタ
制御ユニット
3
14
Di ⊕ F(Rei)
8
保管レジスタ
11
15
Di
17
2
s1
s4
キャッシュ
Kusr
Xef
9
10
71
72
73
18
CPU
26
Permit
Code
Epmp(Kef)
復号部
Kemp
保管レジスタ
復号部
保管レジスタ
5
19
記憶メモリ
Esmpp(Kemp)